

Lecture 2

Algorithm Design

Fundamentals of Computer and Programming

Instructor: Morteza Zakeri, Ph.D. (m-zakeri@live.com)

Spring 2024

Modified Slides from Dr. *Hossein Zeinali* and Dr. *Bahador Bakhshi*

Computer Engineering Department, Amirkabir University of Technology



What We Will Learn

- Sample algorithms to practice problem solving steps
- Input & Output analysis
- Algorithm design
 - Pseudo-code



Algorithm: Average

1. print "Please enter three integers"
2. read x_1, x_2, x_3
3. $\text{sum} \leftarrow x_1 + x_2 + x_3$
4. $\text{average} \leftarrow \text{sum} / 3$
5. print "Average = " average



الگوریتم تشخیص زوج یا فرد بودن عدد

Algorithm: Odd-Even-1

1. print “Please enter an integer”
2. read n
3. $y \leftarrow n \bmod 2$
4. if ($y == 0$)
 print “Number is even”
else
 print “Number is odd”



الگوریتم تشخیص زوج یا فرد بودن عدد

Algorithm: Odd-Even-2

1. print “Please enter an integer”
2. read n
3. if($n < 0$)
 $n \leftarrow -1 * n$
4. while($n \geq 2$)
 $n \leftarrow n - 2$
5. if($n == 0$)
 print “even”
else
 print “odd”

Verify the
Algorithm



الگوریتم تشخیص زوج یا فرد بودن عدد

Algorithm: Odd-Even-3

1. print "Please enter an integer"
2. read n
3. while $(n \geq 2) \text{ or } (n \leq -1)$
 $n \leftarrow n - \text{sign}(n) * 2$
4. if $(n == 1)$
 print "odd"
else
 print "even"



الگوریتمی که يك رشته عدد را که با • تمام می‌شود را می‌گیرد و
تعداد اعداد زوج و فرد را چاپ می‌کند

Algorithm: Count Odd-Even

odd_cnt \leftarrow 0

even_cnt \leftarrow 0

print "Please enter an integer"

read n

while (n != 0)

 y \leftarrow n mod 2

 if (y == 0)

 even_cnt \leftarrow even_cnt + 1

 else

 odd_cnt \leftarrow odd_cnt + 1

 print "Please enter an integer"

 read n

print "Odd = " odd_cnt "Even = " even_cnt



الگوریتمی که يك عدد صحيح مثبت را بگیرد و مجموع ارقام آن را چاپ کند

Algorithm: Digit-Sum

print “Please enter a positive integer”

read n

sum \leftarrow 0

m \leftarrow n

while (n \neq 0)

 y \leftarrow n mod 10

 sum \leftarrow sum + y

 n \leftarrow n - y

 n \leftarrow n / 10

print “sum of digits of” m “ = ” sum

Verify the
Algorithm



الگوریتمی که يك عدد صحيح مثبت را بگیرد و آنرا در مبناي ۸ چاپ کند

Algorithm: Base-8

print "Please enter a positive integer"

read n

$i \leftarrow 0$

while (n \neq 0)

$x[i] \leftarrow n \bmod 8$

$n \leftarrow \text{floor}(n / 8)$

$i \leftarrow i + 1$

$i \leftarrow i - 1$

while (i \geq 0)

print x[i]

$i \leftarrow i - 1$



الگوریتمی که يك عدد صحيح مثبت را بگیرد و فاکتوریل آنرا تولید کند

Algorithm: Factorial-1

print “Please enter a positive integer”

read n

$i \leftarrow 1$

result $\leftarrow 1$

while ($i \leq n$)

 result $\leftarrow i * \text{result}$

$i \leftarrow i + 1$

return result



الگوریتمی که يك عدد صحيح مثبت را بگیرد و فاکتوریل آنرا تولید کند

Algorithm: Factorial-2

print "Please enter a positive integer"

read n

result \leftarrow 1

while (n > 0)

 result \leftarrow result * n

 n \leftarrow n - 1

return result



الگوریتمی که يك عدد صحیح مثبت را بگیرد و فاکتوریل آنرا تولید کند

Algorithm: Factorial-Recursive (n)

```
if (n <= 1)
    return 1
else
    return n * Factorial-Recursive (n - 1)
```



الگوریتمی که يك رشته عدد را که محل عضو اول آن با start و محل عضو آخر آن با end مشخص شده است را به صورت صعودي مرتب کند.

Algorithm: sort (x, start, end)

while (start != end)

$j \leftarrow$ find index of minimum element from start to end

 swap $x[j]$ and $x[start]$

$start \leftarrow start + 1$

=====

Algorithm find_min(x, start, end)

$y \leftarrow start$

$i \leftarrow start + 1$

while ($i \leq end$)

 if ($x[i] < x[y]$)

$y \leftarrow i$

$i \leftarrow i + 1$

return y

	8
	5
	2
	6
	9
	3
	1
	4
	0
	7

Verify the
Algorithm



الگوریتمی که يك رشته عدد را که محل عضو اول آن با start و محل عضو آخر آن با end مشخص شده است را به صورت صعودي مرتب کند.

Algorithm swap(x, j, i)

$\text{temp} \leftarrow x[j]$

$x[j] \leftarrow x[i]$

$x[i] \leftarrow \text{temp}$

Algorithm swap2(x, j, i)

$x[j] \leftarrow x[j] + x[i]$

$x[i] \leftarrow x[j] - x[i]$

$x[j] \leftarrow x[j] - x[i]$



الگوریتمی که آرایه صعودی از اعداد صحیح را بگیرد و آنرا تبدیل به آرایه نزولی کند.

Algorithm reverse(A, start, end)

```
if (start >= end)
    return
else
    swap(A, start, end)
    reverse(A, start + 1, end - 1)
```



Tower of Hanoi

Algorithm `hanoi(n, source, target, auxiliary):`

if $(n \leq 0)$

return

Move $n - 1$ disks from source to auxiliary

`hanoi(n - 1, source, auxiliary, target)`

Move the n th disk from source to target

append the source last disk to the target

Move the $n - 1$ disks that we left on auxiliary onto target

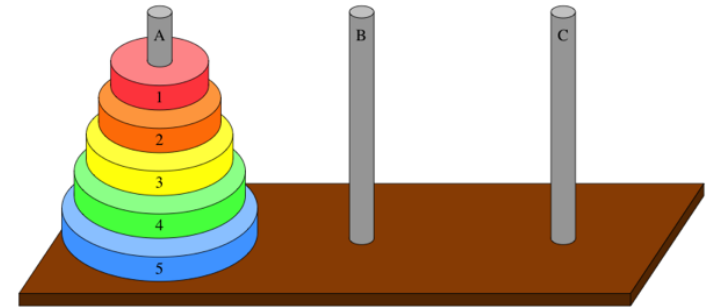
`hanoi(n - 1, auxiliary, target, source)`

$A = [3, 2, 1]$

$B = []$

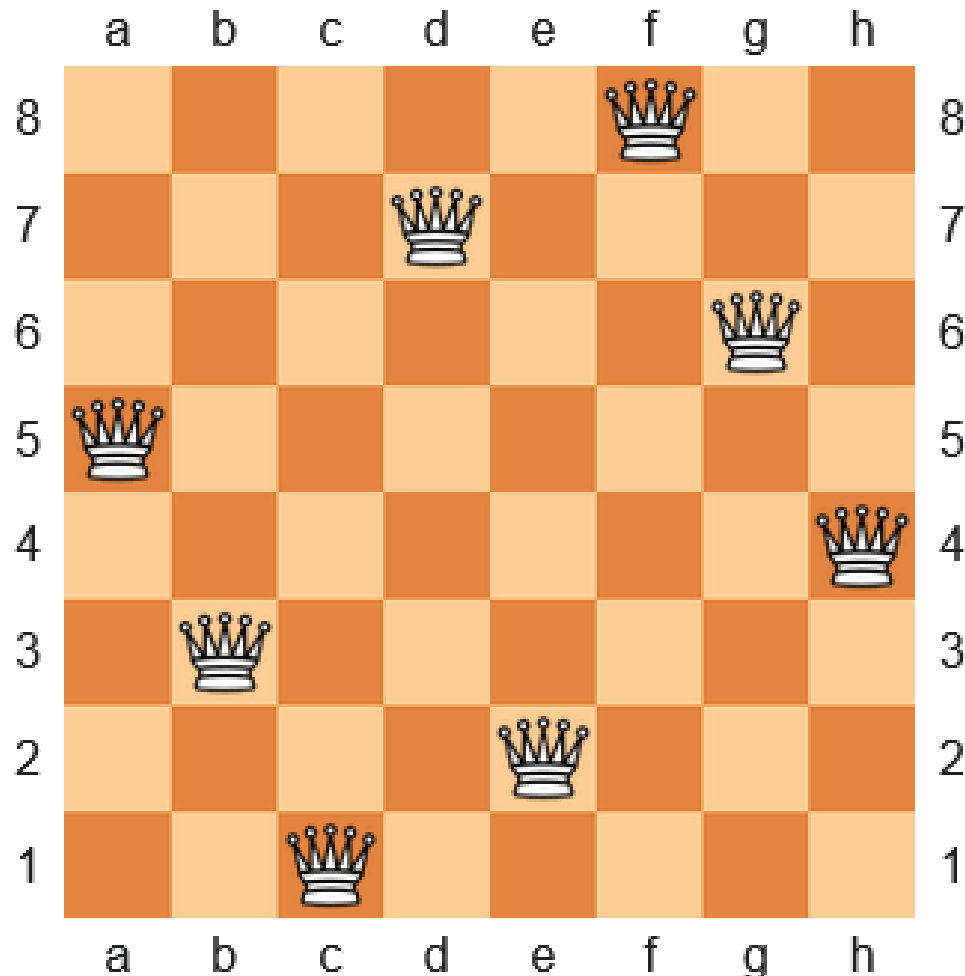
$C = []$

`hanoi(3, A, C, B)`



Eight Queens Puzzle

➤ One possible solution



Summary

- There are more than one algorithm for a problem
 - Efficiency, Complexity, Clarity, ...
- Algorithm (Programming Language) building blocks
 - Calculations (Lecture 4)
 - Input / Output (Lecture 5)
 - Decision Making (Lecture 6)
 - Repeating (Lecture 7)
 - Modular Programming (Lecture 8)
 - Arrays + Memory Management (Lectures 9 + 10)
 - Others (Files, ...) (Lecture 11 + 12)

